PUC Minas

PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS

Laboratório de Algoritmos e Estruturas de Dados – 2/2023

AULA PRÁTICA – ANÁLISE DE COMPLEXIDADE

Prof. Edwaldo Soares Rodrigues

1 – Faça um método que receba um número inteiro n e efetue o número de multiplicações, pedido nos casos a seguir:

a) $5n + 4n^3$
b) $9n^4 + 5n^2 + n/2$
c) $4n^3 + 2$
d) $lg(n) + n^2$
e) $3lg(n) + lg(n)$
f) 2n + 2n2 + lg(n)

2 – Marque verdadeiro ou falso, em cada célula da tabela abaixo:

a)

	$\Theta(1)$	$\Theta(\lg n)$	$\Theta(n)$	$\Theta(n.\lg(n))$	$\Theta(n^2)$	$\Theta(n^3)$	$\Theta(n^5)$	$\Theta(n^{20})$
$f(n) = \lg(n)$	V	V						
$f(n) = n \, . \, \lg(n)$				V				
$f(n) = 5n + 1$			V					
$f(n) = 7n^5 - 3n^2$							V	
$f(n) = 99n^3 - 1000n^2$						V		
$f(n) = n^5 - 99999n^4$							V	

b)

	$\Omega(1)$	$\Omega(\lg n)$	$\Omega(n)$	$\Omega(n.\lg(n))$	$\Omega(n^2)$	$\Omega(n^3)$	$\Omega(n^5)$	$\Omega(n^{20})$
$f(n) = \lg(n)$	V	V						
$f(n) = n \, . \, \lg(n)$	V	V	V	V				
$f(n) = 5n + 1$			V					
$f(n) = 7n^5 - 3n^2$	V	V	V	V	V	V	V	
$f(n) = 99n^3 - 1000n^2$	V	V	V	V	V	V		
$f(n) = n^5 - 99999n^4$	V	V	V	V	V	V	V	

c)

	O(1)	O(lg n)	O(n)	O(n.lg(n))	O(n^2)	O(n^3)	O(n^5)	O(n^{20})
f(n) = lg(n)		V	V	V	V	V	V	V
f(n) = n . lg(n)				V	V	V	V	V
f(n) = 5n + 1			V	V	V	V	V	V
f(n) = $7n^5 - 3n^2$							V	V
f(n) = $99n^3 - 1000n^2$						V	V	V
f(n) = $n^5 - 99999n^4$							V	V

3 – Apresente a função e a taxa de complexidade para as 3 notações vistas em sala, referente ao número de comparações e movimentações de registros, para o pior e melhor caso, para as opções a seguir:

a)

```
void imprimirMaxMin(int [] array, int n){
      int maximo, minimo;

      if (array[0] > array[1]){
            maximo = array[0];      minimo = array[1];
      } else {
            maximo = array[1];      minimo = array[0];
      }

      for (int i = 2; i < n; i++){
            if (array[i] > maximo){
                  maximo = array[i];
            } else if (array[i] < minimo){
                  minimo = array[i];
            }
      }
}
```

Função de complexidade

	MOV	COMP
PIOR	f(n) = 2 + (n-2)	f(n) = 1+ 2(n-2)
MELHOR	f(n) = 2	f(n) = 1 + (n-2)

NOTAÇÕES

	MOV	COMP
PIOR	O(n), Om(n), Th(n)	O(n), Om(n), Th(n)
MELHOR	O(1), Om(1), Th(1)	O(n), Om(n), Th(n)

b)

```
i = 0;

while (i < n) {
      i++;
      a--;
}

if (b > c) {
      i--;
} else {
      i--;
      a--;
}
```

Aqui deveria ser pedido apenas a função de complexidade e as notações, apenas, referente ao número de subtrações.

	função	Notação
PIOR	f(n) = n+2	O(n), Om(n), Th(n)
MELHOR	f(n) = n+1	O(n), Om(n), Th(n)

c)

```
for (i = 0; i < n; i++) {
      for (j = 0; j < n; j++) {
            a--;
            b--;
      }
      c--;
}
```

Aqui deveria ser pedido apenas a função de complexidade e as notações, apenas, referente ao número de subtrações.

	função	Notação
PIOR / MELHOR	f(n) = (2n + 1)n	O(n2), Om(n2), Th(n2)

n2 = n ao quadrado

4 – Apresente o tipo de crescimento que melhor caracteriza as funções abaixo:

	Constante	Linear	Polinomial	Exponencial
3n		X		
1	X			
(3/2)n		X		
$2n^3$			X	
2^n				X
$3n^2$			X	
1000	X			
$(3/2)^n$				X

5 – Classifique as funções $f_1(n) = n.lg(n)$, $f_2(n) = lg(n)$, $f_3(n) = 8n^2$, $f_4(n) = 64$, $f_5(n) = 6n^3$, $f_6(n) = 8^{2n}$ e f7(n) = 4n de acordo com o crescimento, do mais rápido para o mais lento.

Classificação das funções de complexidades, em ordem de eficiência. Quanto antes, mais eficientes os algoritmos representados pelas funções
F4
F2
F7
F1
F3
F5
F6